FINAL TECHNICAL REPORT:
Award Number G13AP00020
The Mechanics of Episodic Tremor and Slip with Implications for Seismic Hazards in Cascadia

November 11, 2014

Paul Segall
(650-723-7241, segall@stanford.edu)
Geophysics Department
Stanford University
Stanford, CA 94305-2215

Award Dates: 04/01/2013 – 3/31/2014

Abstract

We continue development of physical models of slow slip events, with a focus on understanding the implications for seismic hazards in the Pacific Northwest. In previous reports we described physics based models of slow slip events (SSE) resulting from dilatant strengthening, 2D numerical simulations that showed how SSE could potentially evolve into damaging dynamic ruptures, compared dilatant strenghting models to those involving a change from rate-weakening to rate-strengthening friction, and analyzed geodetic data (GPS, leveling, and tide-gauge) to constrain the spatial distribution of fault locking and to compare with predictions of the physics-based models.

Previous physics-based models have been restricted to 2D numerical simulations. In this report we focus on work done to accelerate 3D numerical simulations of fault behavior. We completed development of Hierarchical-matrix (H-matrix) compression software library `hmmvp`, that allows computation of Boundary Element calculations in $O(n \log n)$ time, rather than $O(n^2)$ time for uncompressed matrix vector products. We also developed a method we call IGA that allows for modeling with non-uniform meshes without sacrificing accuracy. Non-uniform meshes are highly desirable when either fault properties or background normal stresses are spatially variable, such that critical frictional length scales are spatially variable. To illustrate the benefits of our methods, we consider a 3D simulation of a slow-slip event that grows into a (quasi-) dynamic rupture. The nonuniform mesh has 1.2 million elements; the corresponding uniform mesh would have 16.8 million, or 14.5× more than the non-uniform mesh. The H-matrix for the nonuniform mesh at error tolerance $\varepsilon = 10^{-5}$ is 6.6 GB (single-precision). The H-matrix for the uniform mesh would be 95.6 GB. The full BEM matrix for the nonuniform mesh would require 4.9 TB (terabyte) and the uniform mesh 1.0 PB (petabyte). In summary, the combination of IGA and H-matrix compression allows well resolved problems to be run on standard desktop computers. A paper describing the methods has now been published in

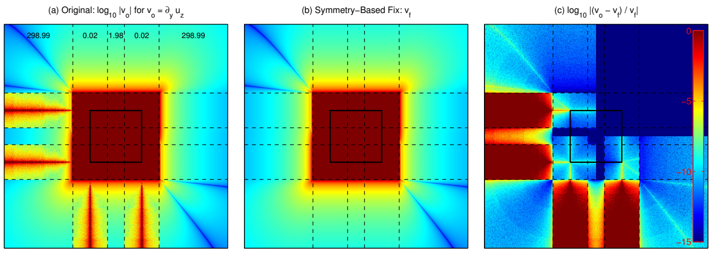
Bradley [2014], and the software is open source and available at `https://pangea.stanford.edu/research/CDFM/software/index.html`.

1 Report

Simulation of ETS events involves combining the equations of elasticity with appropriate fault constitutive laws. Slip rate is determined by friction and tractions acting on faults. Each fault is discretized into N elements. Each component of slip $s^{(i)}$ and traction $\tau^{(j)}$ is related by a matrix-vector product (MVP). In previous reports we have described $\mathcal{H}$-matrix compression methods that allow the MVP to be computed in $O(NlogN)$ operations, as opposed to $O(N^2)$ for conventional MVP.

The number of fault elements, N is determined by the need to resolve the smallest relevant spatial scales. In fault simulations $Gd_c/b(\sigma - p^{\infty})$, controls the shape of the rupture front. Here $G, (\sigma - p^{\infty})$ are shear modulus and effective stress, d_c, b from friction law. For laboratory values of d_c, N^2 can become prohibitively large. For most problems of interest frictional properties vary spatially (e.g., transition from velocity-strengthening to weakening friction, 'asperities' with different $a - b$, or $(\sigma - p^{\infty})$, low frequency earthquake sources, etc). These considerations argue strongly for a non-uniform fault mesh.

As shown below, constant-slip elements (CSE) generate intolerable errors on a nonuniform mesh. We have developed a method we call IGA (interpolate, multiply by the Green's function array, average) that greatly suppresses these artifacts. It is implemented in the library `dc3dm`. The user specifies a *resolution function* $f_r(x, y)$ that maps fault location (x, y) to desired element size. For example, $f_r(x, y) \propto d_c(x, y)/b(x, y)\sigma(x, y)$. `dc3dm` generates a nonuniform mesh, computes the $\mathcal{H}$-matrix approximation to the array of Green's functions G, and provides an interface to the MVP with G.

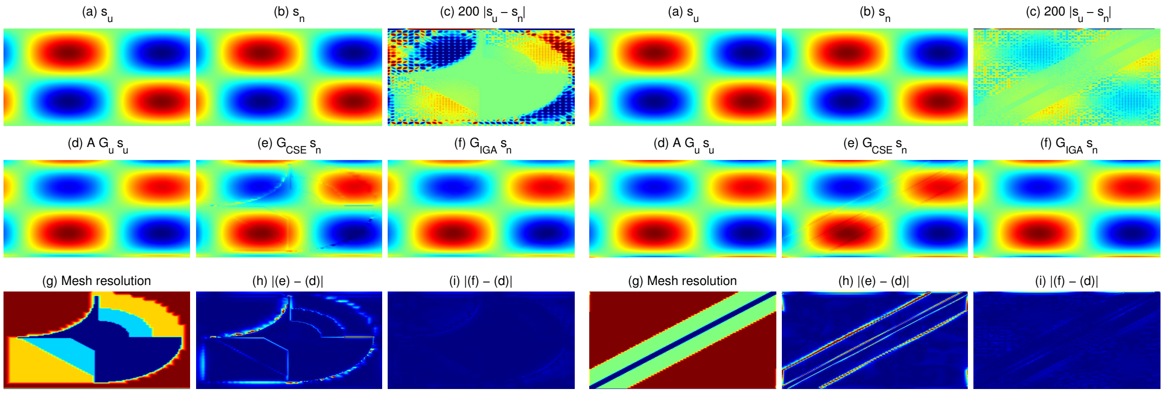


(a) (b) (c)

Figure 1: Error and fix to `dc3d` [Okada, 1992]. Non-uniform mesh with skinny strips aligned with the rectangle's sides are indicated by dashed lines with width 0.02 units. The area outside the rectangle and strips is ~ 300 units. Dip, strike, and opening dislocation are all one unit. a) original Okada [1992] code, b) applying the symmetry fix, and c) relative difference between (a) and (b).

We use the Fortran code `dc3.f` [Okada, 1992] to compute Green's functions for constant-slip rectangular dislocations in an elastic half-space. We first addressed a numerical error in `dc3d` that negatively impacted our high-resolution BEM applications. Fig. 1(a) shows $\log_{10} |\partial_y u_z|$, in the plane of, and due to, the 2-unit rectangular source outlined by the solid black rectangle. The mesh is highly nonuniform to emphasize the problem. There are 4 cones (out of possible 8) of numerical error; each emerges from an edge of the rectangular dislocation. The cause of the error is numerical cancellation in expressions of the form $R + y$ for $y = \eta < 0$ or $y = \xi < 0$, where $R = (\xi^2 + \eta^2 + q^2)^{1/2}$ and ξ, η, q are local coordinate directions (notation of Okada [1992]). The solution is to use the symmetry of the problem to reflect the source-receiver geometry across the q-ξ and q-η planes so that the transformed receiver is in the $\xi, \eta > 0$ quadrant, where the cancellation error does not occur. Then reflect the transformed solution back to the original space. The result is shown in Fig. 1(b), and the point-wise relative differences between (a) and (b) in Fig. 1(c). This fix is particularly important for `dc3dm` because the nonuniform mesh aligns large rectangle centers with small rectangle edges, the situation in which error is greatest. We distribute a corrected version of `dc3.f` (`dc3omp.f`) with `disloc3d` (including OpenMP version) on our group webpage.

Mathematics of IGA. The standard constant-slip elements (CSE) method collocates slip and traction at element centers. We refer to the associated array of Green's functions as G_{CSE}. Constant-slip elements cause error where two adjacent elements have different sizes because of the singularity at the element edges. (On a uniform mesh the singularities cancel each other.) The solution is to increase the order of the approximation. We choose a nonuniform mesh with the *tiling* property that the smallest element divides every other element in the mesh. Associate with this mesh a uniform mesh whose element size is the smallest in the nonuniform mesh. We call the small elements that tile a large element in the nonuniform mesh *subelements* and the large element the *superelement*. Let $I_{n\to u}$ interpolate a quantity on the nonuniform (n) mesh to the uniform (u) one. Define G_u as the array of Green's functions for the uniform mesh; we use subscript u and n throughout. Finally, let $A_{u\to n}$ be the linear operator that averages quantities defined on subelements to the superelements. The IGA method defines $G_{\text{IGA}} \equiv A_{u\to n} G_u I_{n\to u}$. Then $\tau_u = G_u s_u$ and $\tau_n = G_{\text{IGA}} s_n = A_{u\to n} G_u I_{n\to u} s_n$.



(a) (b)

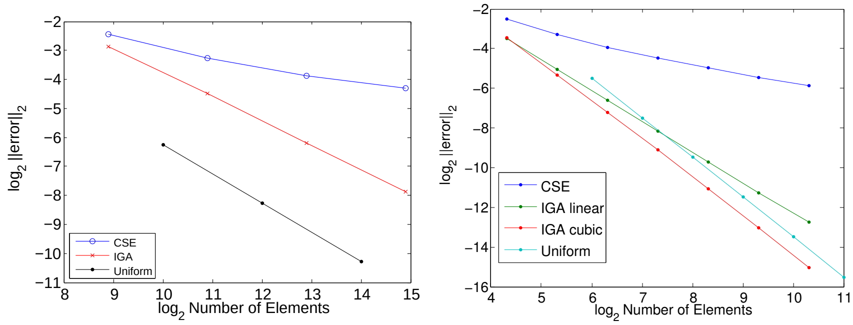
Figure 2: Matrix-vector product test of IGA. In each of the two examples shown: a) and b) are test slip functions on uniform and non-uniform meshes, respectively. c) scaled absolute difference. d) – f) show the tractions on the uniform mesh, the naive approach to the non-uniform mesh, and computed according to IGA, respectively. h) and i) show the error in the naive approach and with IGA at the same color scale. See text for details.

Note that the operator $I_{n\to u}$ is not trivial to define. The first step to is to discretize the rectangular domain D, by dividing into n_D subdomains, square to within a tolerance (1.1 is good). Then each square is refined according to f_r using an adaptive quadtree. This yields nearly square elements and quadtree mesh (QTM) that matches the resolution function. Interpolation is based on triangulation of the QTM element centers; details are omitted for brevity. Periodic, velocity, and free-surface boundary conditions are allowed.

So far we have described $I_{n\to u}$, which is associated with G_u. In practice we do not use G_u, for the size of G_u grows with the smallest element in the nonuniform mesh. Instead, *approximate*

IGA (AIGA) associates a neighborhood with each receiver element. Associated with each neighborhood is a smallest element – the largest subelement that tiles the superelements in the neighborhood. This is the crucial aspect of AIGA that makes computations associated with it globally independent of the smallest element size in the nonuniform mesh. There is an operator I^i associated with each neighborhood, where i indexes the subelement sizes. (There are generally on the order of 1 to 10 subelement sizes, which of course is far fewer than the number of elements N.) Mathematically, the elements of G_{AIGA} are formed by obtaining elements from a family of IGA arrays $G^i_{\text{IGA}} \equiv A_{u^i \to n} G_{u^i} I_{n \to u^i}$ according to the neighborhoods. We omit a description of the rules to define a neighborhood.

Building the QTM and the associated interpolant take a few tens of seconds for $N \sim 10^6$ elements. Moreover, computing time scales linearly in N. Thus, all mesh preprocessing is a negligible part of constructing G_{AIGA}. H-matrix construction scales approximately linearly in N. Profiling shows that about 90% of CPU time spent in providing data to `hmmvp`'s compression program is spent in the Green's function routine, roughly independent of mesh. Hence all bookkeeping operations associated with the QTM and A and I operators are also essentially negligible.

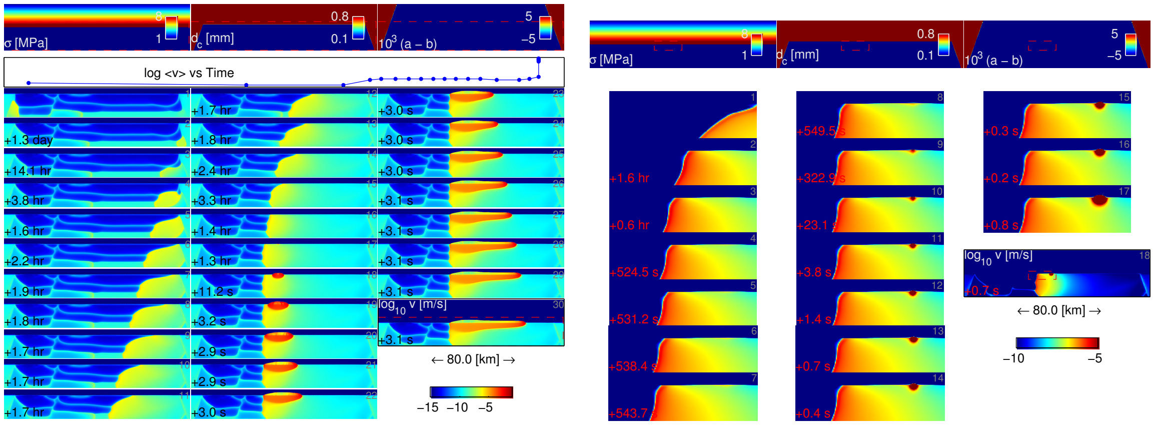


(a) (b)

Figure 3: Convergence tests of IGA. **(a)** 3D with linear interpolation in IGA. **(b)** 2D with linear and cubic interpolation in IGA.

Fig. 2(a,b) show matrix-vector product tests ($\tau = Gs$) with smooth test functions s for two different nonuniform meshes for the same domain D. D has periodic boundary conditions on the west and east sides, a velocity boundary condition on the south, and a free surface boundary condition on the north. Each figure has nine subplots: (a) is s_u, the test function on the uniform mesh. (b) is s_n, the test function on the nonuniform mesh. (c) shows the absolute difference between the two, scaled by 200. (d) shows $AG_u s_u$, which is τ_u projected (by averaging within super elements) onto the nonuniform mesh. (e) is $G_{\text{CSE}} s_n$, the shear stress calculated by the standard CSE method. (f) is $G_{\text{IGA}} s_n$, the solution calculated by IGA. (d)–(f) are on the same color scale. (g) shows mesh element sizes. There are four element sizes, with areas relative to the smallest element of 1, 4, 16, 64. (h) and (i) are on the same color scale and show error in the CSE and IGA solutions. The key observation is that the CSE method has spikes in its error field at the interfaces of different element sizes, while the IGA solution does not.

We verify IGA and `dc3dm` by comparing with fine uniform-mesh solutions. Fig. 3(a) shows convergence tests based on test functions and meshes like those in Fig. 2(b). Solutions are compared with that on the finest uniform mesh. IGA (red) achieves almost the same convergence rate as the uniform-mesh method (black), while the CSE method's (blue) *convergence rate* is greatly slowed by errors at the element-size transitions. Please note that the convergence rate is the slope of the curves shown in Fig. 3. We believe higher-order interpolation will give IGA the same convergence as the uniform-mesh. We tested this in 2D (1D fault) as in Fig. 3(b). When cubic, rather than linear, interpolation is used, IGA's convergence rate matches that of the uniform-mesh.



(a)

(b)

Figure 4: $\log_{10}$ slip speed. Numbers with + prefix indicate elapsed time since the previous image. Time progresses down the columns. Average slip speed *vs* time is plotted for reference. Properties are shown at the top. Red dashed box shows zoomed-in portion.

We present a preliminary 3D (2D fault) simulation that illustrates the temporal relationship between the ultimate SSE and a dynamic rupture (Fig. 4). Properties are shown at the top of (a). For meshing reasons, we artificially diminished $\bar{\sigma}$ at shallower depth, but will eliminate this in the future. In Fig. 4(a) the ultimate SSE nucleates in the right corner and propagates left. Midway across, an earthquake nucleates at the $\bar{\sigma}$ transition. Fig. 4(b) zooms in on the nucleating earthquake. Note first that the SSE triggers the dynamic event, as in the 2D simulations of Segall and Bradley [2012]. Second, the DE first propagates along strike in the reverse direction of the SSE. (The end of the DE has not been simulated yet.) We note that the dynamic event does not rupture updip beyond the d_c transition, which was used to limit the size of the mesh, indicating more work needs to be done.

To illustrate the impact of `hmmvp` and `dc3dm` consider that the nonuniform mesh has 1.2 million elements; the corresponding uniform mesh would have 16.8 million, or 14.5 times more. The H-matrix for the nonuniform mesh at error tolerance $\varepsilon = 10^{-5}$ is 6.6 GB (single-precision). This fits comfortably in the main memory of a modest off-the-shelf computer. The H-matrix for the uniform mesh would be 95.6 GB. A uniform mesh allows one to exploit the FFT along strike; this would require 128.0 GB storage. The full BEM matrix for the nonuniform mesh would require 4.9 TB (terabyte) and the uniform mesh 1.0 PB (petabyte). The two numbers of greatest interest are those for the uniform-mesh FFT implementation (128.0 GB) and our implementation (6.6 GB). The factor 19.4 difference in memory and also roughly in MVP time means a single modest shared-memory computer can handle this problem, as opposed to requiring a computer cluster.

2 Bibliography supported this reporting period only

1. A. M. Bradley. dc3dm: Software for efficient quasistatic dislocation-traction operators on nonuniformly discretized rectangular faults. Abstract T51D-2490 presented at 2013 Fall Meeting, AGU, San Francisco, Calif., 3-7 Dec., 2013a.

2. A. M. Bradley. dc3dm: Software for efficient quasistatic dislocation-traction operators on nonuniformly discretized rectangular faults. Poster at 2013 SCEC Annual Meeting, 2013b.

3. A. M. Bradley. Software to efficiently form and apply quasistatic dislocation-traction operators. *Seismological Research Letters, 85(6)*, doi:10.1785/0220140092, 2014.

References

A. M. Bradley. dc3dm: Software for efficient quasistatic dislocation-traction operators on nonuniformly discretized rectangular faults. Abstract T51D-2490 presented at 2013 Fall Meeting, AGU, San Francisco, Calif., 3-7 Dec., 2013a.

A. M. Bradley. dc3dm: Software for efficient quasistatic dislocation-traction operators on nonuniformly discretized rectangular faults. Poster at 2013 SCEC Annual Meeting, 2013b.

A. M. Bradley. Software to efficiently form and apply quasistatic dislocation-traction operators. *To be submitted to BSSA*, 2014.

Y. Okada. Internal deformation due to shear and tensile faults in a half-space. *Bull. Seism. Soc. Am.*, 82, 1992.

P. Segall and A. M. Bradley. Slow-slip evolves into megathrust earthquakes in 2d numerical simulations. *Geophysical Research Letters*, 39(18):L18308, 2012. doi: 10.1029/2012GL052811.